

# On Meshfree GFDM Solvers for the Incompressible Navier–Stokes Equations

Pratik Suchde<sup>a,b,\*</sup>, Jörg Kuhnert<sup>a</sup>, Sudarshan Tiwari<sup>b</sup>

<sup>a</sup>*Fraunhofer ITWM, 67663 Kaiserslautern, Germany*

<sup>b</sup>*Department of Mathematics, University of Kaiserslautern, 67663 Kaiserslautern, Germany*

---

## Abstract

Meshfree solution schemes for the incompressible Navier–Stokes equations are usually based on algorithms commonly used in finite volume methods, such as projection methods, SIMPLE and PISO algorithms. However, drawbacks of these algorithms that are specific to meshfree methods have often been overlooked. In this paper, we study the drawbacks of conventionally used meshfree Generalized Finite Difference Method (GFDM) schemes for Lagrangian incompressible Navier–Stokes equations, both operator splitting schemes and monolithic schemes. The major drawback of most of these schemes is inaccurate local approximations to the mass conservation condition. Further, we propose a new modification of a commonly used monolithic scheme that overcomes these problems and shows a better approximation for the velocity divergence condition. We then perform a numerical comparison which shows the new monolithic scheme to be more accurate than existing schemes.

**Keywords:** Meshfree, Particle method, Finite difference, Navier–Stokes, Incompressible flow, Hydrodynamics, FPM, Finite Pointset Method

---

## 1. Introduction

In hydrodynamic simulations, the computational domain is often very complex or is rapidly changing with time. For such simulation domains, the task of meshing can prove to be very challenging. Meshfree or meshless methods are becoming increasingly popular for such simulations to avoid the tasks of meshing and remeshing. These methods use the numerical basis of a set of arbitrarily distributed nodes without any underlying mesh to connect them. These nodes could either be mass carrying particles or numerical points. For each node, the only connectivity information required is a local set of neighbouring nodes over which approximations are carried out.

Meshfree Generalized Finite Difference Methods (GFDMs) are one such class of meshfree methods. Meshfree GFDMs are strong-form methods based on weighted least squares approximations. They have been widely used (for example, [13, 20, 36, 44]) and are referred under various names, including FPM, which stands for both the Finite Pointset Method [38] and the Finite Point Method [30]; the Kinetic Meshless Method (KMM) [32] and the Least Squares Kinetic Upwind Method (LSKUM) [14]. They provide several advantages over meshfree particle methods such as Smoothed Particle Hydrodynamics (SPH) [24]. They provide a framework to naturally incorporate boundary conditions which is not possible in SPH. They are more adaptive in terms of the spatial discretization since they use numerical approximation points instead of mass carrying particles. These advantages, however, come at the price of a lack of strict conservation.

Several mesh-based algorithms to solve the incompressible Navier–Stokes equations have been extended to meshfree methods. These include operator splitting methods such as the projection method [8], SIMPLE

---

\*Corresponding author

Email address: [pratik.suchde@itwm.fraunhofer.de](mailto:pratik.suchde@itwm.fraunhofer.de) (Pratik Suchde)

[31] and PISO [18]. However, drawbacks of these algorithms that are specific to meshfree methods have often been overlooked. One important example of such a drawback is that the discrete Laplace operator is not the same as the discrete divergence of the discrete gradient operator. This results in inaccurate approximations to the mass conservation equation. While this problem has been solved in the context of Finite Volume Methods (FVMs) with staggered grids, this problem persists in meshfree methods and introduces errors in most operator splitting algorithms.

In this paper, we present a few commonly used meshfree solvers, both operator splitting solvers and monolithic solvers, for the incompressible Navier–Stokes equations and study the drawbacks of these algorithms specific to meshfree GFDMs. We then propose a new monolithic solver which attempts to overcome these drawbacks. This is done by solving an over-determined problem. The mass and momentum conservation equations are solved together, and simultaneously with a pressure-Poisson equation which is needed to improve stability. All the methods considered are spatially second order accurate and use first order time integrations. However, the new monolithic method proposed here is numerically shown to be much more accurate than existing methods. This is as a consequence of the new method providing better local approximations to the mass conservation condition.

In our earlier paper [37], we presented a method to improve global conservative properties in meshfree GFDMs, with an application to the incompressible Navier–Stokes equations. That was done by introducing an approximate discrete divergence theorem, without changing the local accuracy of the conservation equations. In contrast, in this paper, we present a method to improve the local accuracy of the mass conservation equation.

The paper is organized as follows. In Section 2, we give an overview of two variants of meshfree GFDMs. In Section 3, we present existing meshfree GFDM algorithms used to numerically solve the incompressible Navier–Stokes equations and analyze their drawbacks. A new coupled velocity-pressure solver which addresses these drawbacks is presented in Section 4. Numerical results and comparisons of the new method with existing methods are presented in Section 5, and the paper is concluded with a discussion in Section 6.

## 2. Meshfree Finite Differences

In meshfree GFDMs, a cloud of  $N$  numerical points is used to discretize the computational domain  $\Omega$ . These points, with positions  $\vec{x}_i$ ,  $i = 1, \dots, N$ , are usually irregularly spaced and include both points in the interior and on the boundary  $\partial\Omega$  of the domain. The only connectivity information required for each point  $i$  is a set of neighbouring points  $S_i$ . In most cases, this neighbourhood or support  $S_i$  consists of the  $n = n(i)$  points closest to it,  $S_i = \{\vec{x}_j : \|\vec{x}_j - \vec{x}_i\| \leq h\}$ , where  $h = h(\vec{x}, t)$  is the smoothing length, which determines the spatial discretization size. In addition to the smoothing length  $h$ , the point cloud is also described by the parameters  $r_{max}$  and  $r_{min}$ . Initial set up of the point cloud, and local addition and deletion of points during a simulation is done such that there exists at least one point in every sphere of radius  $r_{max}h$ , and that no two points are closer than  $r_{min}h$ . Further details about point cloud organization and management can be found in [10, 19, 34].

There are two variations of meshfree finite differences that we consider in this paper, both of which are based on moving least squares approximations. The first is the widely used approach based on the work of Liszka and Orkisz [23], which we refer to as the classical GFDM. This method very closely resembles traditional finite differences, and is obtained by minimizing errors obtained from Taylor expansions. The second approach is a modification of the classical GFDM based on the work of Tiwari and Kuhnert [38] in which the errors in the PDE considered is minimized simultaneously with the errors obtained from Taylor expansions. A comparison between these approaches has been done by Illiev and Tiwari [17]. In both variations, we consider Taylor expansions up to second order terms. Higher order accuracy can be attained in both formulations and have been considered by Milewski [27].

### 2.1. Classical meshfree GFDM

This classical method is based on the work of Liszka and Orkisz [23]. For a function  $u$  defined at each numerical point  $i = 1, 2, \dots, N$ , its derivatives are approximated as

$$\partial^* u(\vec{x}_i) \approx \tilde{\partial}_i^* u = \sum_{j \in S_i} c_{ij}^* u_j, \quad (1)$$

where  $*$  =  $x, y, xx, \Delta$ , etc. represents the differential operator being approximated,  $\partial^*$  represents the continuous  $*$ -derivative, and  $\tilde{\partial}_i^*$  represents the discrete derivative at point  $i$ . For a point  $i$ , consider Taylor expansions around it at each neighbouring point  $j \in S_i$

$$e_j + u(\vec{x}_j) = u(\vec{x}_i) + \nabla u \cdot (\vec{x}_j - \vec{x}_i) + \frac{1}{2}(\vec{x}_j - \vec{x}_i)^T D(\vec{x}_j - \vec{x}_i). \quad (2)$$

The unknown coefficients  $[\nabla u, D]^T$  are computed by a weighted least squares method, by minimizing

$$\min J = \sum_{j \in S_i} W_j e_j^2, \quad (3)$$

where  $W$  is a weighting function used to make sure that the points closer to the central point  $i$  have a larger impact than the points farther away. It is usually taken as a Gaussian distribution

$$W_j = \exp\left(-\alpha \frac{\|\vec{x}_j - \vec{x}_i\|^2}{h^2}\right), \quad (4)$$

where  $\alpha$  is a positive constant. Note that the weighting function is only defined on the local support  $S_i$  consisting of  $n(i)$  points. For the sake of brevity, we present only the case of one spatial dimension. Eq. (2) and Eq. (3) lead to the following system which is solved at each point  $i = 1, \dots, N$

$$\begin{pmatrix} e_1 \\ \vdots \\ e_n \end{pmatrix} = \begin{pmatrix} \delta x_1 & \frac{1}{2}\delta x_1^2 \\ \vdots & \vdots \\ \delta x_n & \frac{1}{2}\delta x_n^2 \end{pmatrix} \begin{pmatrix} u_x \\ u_{xx} \end{pmatrix} - \begin{pmatrix} u_1 - u_i \\ \vdots \\ u_n - u_i \end{pmatrix}. \quad (5)$$

where  $\delta x_j = x_j - x_i$ . Or, in short form  $\vec{E} = M\vec{a} - \vec{b}$ , with  $\vec{a} = (u_x, u_{xx})^T$  and  $\vec{b} = (u_1 - u_i, \dots, u_n - u_i)^T$ . The minimization Eq. (3) can be rewritten as

$$\min J = (M\vec{a} - \vec{b})^T W (M\vec{a} - \vec{b}), \quad (6)$$

where  $W$  is a diagonal matrix with entries  $W_1, \dots, W_n$ . A formal minimization leads to

$$\vec{a} = [(M^T W M)^{-1} M^T W] \vec{b}. \quad (7)$$

Which leads to the differential operator stencils

$$u_x \Big|_i = \sum_{j \in S_i} c_{ij}^x (u_j - u_i), \quad (8)$$

$$u_{xx} \Big|_i = \sum_{j \in S_i} c_{ij}^{xx} (u_j - u_i), \quad (9)$$

where  $c_{ij}^x$  and  $c_{ij}^{xx}$  represent the values in the first and second row respectively of the matrix  $[(M^T W M)^{-1} M^T W]$  in Eq. (7). These stencils are then used to solve the PDE. For example, if we consider the PDE

$$au + bu_x + cu_{xx} = d. \quad (10)$$

The derivative approximations Eq. (8), Eq. (9) are substituted into the PDE to obtain

$$au_i + b \sum_{j \in S_i} c_{ij}^x (u_j - u_i) + c \sum_{j \in S_i} c_{ij}^{xx} (u_j - u_i) = d, \quad i = 1, \dots, N \quad (11)$$

which forms a large sparse implicit system which is solved with an iterative method.

An alternate, but equivalent, way to arrive at the stencil coefficients in Eq. (8) is to ensure that the derivatives of monomials  $m \in \mathcal{M}$ , up to the order of accuracy desired, are exactly reproduced [34]

$$\sum_{j \in S_i} c_{ij}^x m_j = \partial_i^x m \quad \forall m \in \mathcal{M}, \quad (12)$$

$$\min J = \sum_{j \in S_i} W_{ij} (c_{ij}^x)^2. \quad (13)$$

To improve stability and to improve accuracy for Dirichlet boundary conditions, a Kronecker-delta property is often fulfilled, either by the addition of a Kronecker-delta function  $\delta_{ij}$  to the monomial set  $\mathcal{M}$  [22] or through the weighting function  $W$  [28].

## 2.2. Tiwari Approach

This method is based on the work of Tiwari and Kuhnert [38]. In this method, stencil coefficients are found that include the PDE being solved as a local constraint. Then a function is found that satisfies the conditions on all the stencils computed. The least squares procedure for minimizing the errors includes not just the Taylor expansions, but also the error in solving the PDE itself. Consider the PDE Eq. (10) used in the previous section. We proceed in the same way as done in Section 2.1. The system Eq. (5) and Eq. (3) get extended to

$$\begin{pmatrix} e_1 \\ \vdots \\ e_n \\ e_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & \delta x_1 & \frac{1}{2} \delta x_1^2 \\ \vdots & \vdots & \vdots \\ 1 & \delta x_n & \frac{1}{2} \delta x_n^2 \\ a & b & c \end{pmatrix} \begin{pmatrix} u \\ u_x \\ u_{xx} \end{pmatrix} - \begin{pmatrix} u_1 \\ \vdots \\ u_n \\ d \end{pmatrix}, \quad (14)$$

$$\min J = \sum_{j \in S_i} W_j e_j^2 + W_{n+1} e_{n+1}^2, \quad (15)$$

which is solved at each point  $i = 1, \dots, N$ . The stencils to the function approximation of  $u$  are the only ones of importance. Proceeding in the same way as done in Section 2.1 leads to a system similar to that obtained in Eq. (7), the first row of which gives the function approximation stencils

$$u_i = \sum_{j \in S_i} \alpha_{ij} u_j + \beta_i d. \quad (16)$$

We then find a function  $u$  that satisfies all these stencils by solving a large sparse implicit system with an iterative method.

$$(1 - \alpha_{ii})u_i - \sum_{\substack{j \in S_i \\ j \neq i}} \alpha_{ij} u_j = \beta_i d. \quad i = 1, \dots, N. \quad (17)$$

This results in a diagonally dominant system. One of the advantages of this direct approach over the classical GFDM one is that it can be used to solve algebraically over-determined, but well-posed, systems. i.e. systems with more PDEs than variables. An important use of this is to solve PDE systems on boundary points along with the imposed boundary conditions, which is not possible by the classical GFDM. Tiwari and Kuhnert [38] used this framework for pressure-Poisson equations in Navier–Stokes solvers. There, an over-determined system was solved on boundary points, consisting of both the boundary conditions and the pressure-Poisson equation itself. The same idea was later extended for imposing free surface boundary

conditions [39], for compressible flows [21], and recently for heat transfer boundary conditions by Reséndiz and Saucedo [33], among other applications. A similar idea of adding the PDE on boundaries was also done in context of the meshfree Radial Basis Functions (RBF), by using an additional set of nodes adjacent to the boundary [12], which was shown to increase accuracy over standard RBF. Using Tiwari's approach, a similar increase in accuracy is obtained in meshfree GFDMs and can be done without the addition of extra nodes.

However, other than solving a PDE along with the boundary condition at boundary points, to the best of our knowledge, this framework has not been used to solve over-determined problems across all points in the computational domain. In this paper, we show how this framework could be used to devise a new algorithm which reduces some problems with existing meshfree GFDM solvers for the incompressible Navier–Stokes equations by solving an over-determined problem across the entire computational domain.

### 3. Existing meshfree GFDM algorithms for the incompressible Navier–Stokes equations

We consider the incompressible Navier–Stokes equations in Lagrangian form.

$$\frac{D\vec{x}}{Dt} = \vec{v}, \quad (18)$$

$$\nabla \cdot \vec{v} = 0, \quad (19)$$

$$\frac{D\vec{v}}{Dt} = \frac{\eta}{\rho} \Delta \vec{v} - \frac{1}{\rho} \nabla p + \vec{g}, \quad (20)$$

where  $\vec{v}$  is the fluid velocity,  $p$  is the pressure,  $\rho$  is the density,  $\eta$  is the dynamic viscosity and  $\vec{g}$  includes both gravitational acceleration and body forces. All the algorithms considered below start with an update of point locations by solving Eq. (18) according to

$$\vec{x}_i^{(n+1)} = \vec{x}_i^{(n)} + \vec{v}_i^{(n)} \Delta t + \frac{\vec{v}_i^{(n)} - \vec{v}_i^{(n-1)}}{\Delta t} (\Delta t)^2, \quad (21)$$

for each point  $i = 1, \dots, N$ , where the bracketed superscript refers to the time level. Following the movement of points, mass conservation Eq. (19) and momentum conservation Eq. (20) are solved according to one the methods mentioned below.

Two broad classes of algorithms are used to solve the incompressible Navier–Stokes equations. Namely, operator splitting methods, and monolithic methods. In operator splitting methods, also referred to as fractional step methods (FSM), partitioned methods, or pressure-segregation methods, the momentum conservation and mass conservation equations are solved in two separate steps. In monolithic methods, also referred to as coupled velocity-pressure methods, the mass conservation and momentum conservation equations are solved together in one large system. In the remainder of this section, we present existing meshfree methods in both these classes and study their drawbacks. In the next section, we present a new monolithic meshfree method which helps reduce these problems in existing solvers.

#### 3.1. Operator Splitting Methods

Projection methods are a type of operator splitting methods, based on the Helmholtz-Hodge decomposition [5], which have been widely used for approximating incompressible and weakly compressible fluid flows in Finite Volume Methods [1, 7]. These method have also been commonly used in meshfree contexts (see, for example, [39, 42]). They consist of two steps. In the first step, an intermediate velocity is computed by solving the momentum conservation equation. This intermediate velocity is then projected to a divergence-free field with the help of a correction pressure to obtain the final velocity. The intermediate velocity  $\vec{v}^*$  is obtained by solving

$$\frac{\vec{v}^* - \vec{v}^{(n)}}{\Delta t} = \frac{\eta}{\rho} \Delta \vec{v}^* - \frac{1}{\rho} \nabla p^* + \vec{g}, \quad (22)$$

where  $p^*$  is a pressure guess. In the original projection method of Chorin [8], the pressure guess is taken to be zero. Non-zero values of this pressure guess have been shown to produce more accurate results [7]. Throughout this paper, we take the pressure guess to be the pressure at the previous time-step  $p^* = p^{(n)}$ . The step of computation of the intermediate velocity is often done explicitly, by replacing the  $\Delta \vec{v}^*$  term in Eq. (22) with  $\Delta \vec{v}^{(n)}$  [39]. However, the implicit way used in Eq. (22) produces more accurate, and often more stable, results.

In the second step, the velocity is corrected by projecting it to a divergence free space by

$$\vec{v}^{(n+1)} = \vec{v}^* - \frac{\Delta t}{\rho} \nabla p_{corr}. \quad (23)$$

The pressure correction  $p_{corr}$  is computed by a pressure-Poisson equation obtained by applying the divergence operator to Eq. (23) and setting  $\nabla \cdot \vec{v}^{(n+1)} = 0$

$$\frac{\Delta t}{\rho} \Delta p_{corr} = \nabla \cdot \vec{v}^*. \quad (24)$$

Finally the pressure is updated by

$$p^{(n+1)} = p^* + p_{corr}. \quad (25)$$

Different variations of such projection methods including higher order time integration have been studied, for example, by Brown *et al.* [7]. A variety of boundary conditions have been used in these methods. A discussion on boundary conditions for projection methods in the mesh-based context can be found in Denaro [25]. In meshfree projection methods, a common approach is to obtain the boundary condition by projecting the underlying equation solved at interior points on the outward facing unit normal. Such approaches and the required stabilization can be found in Fang and Parriaux [11] and Boroomand *et al.* [6]. In this paper, we use boundary conditions dependent on the physics of the underlying problem as done in Seibold [34]. Further, for the spatial discretization of Eq. (22) and Eq. (24) we use the classical meshfree GFDM presented in Section 2.1, as done, for example, by Drumm *et al.* [10]. We now consider a few drawbacks of such projection methods in the meshfree context.

#### *Consistency of numerical differential operators:*

While applying the divergence operator to Eq. (23) to obtain Eq. (24), the assumption is made that the Laplace operator is the same as the divergence of the gradient operator. Specifically,

$$\nabla \cdot \nabla p_{corr} = \Delta p_{corr}. \quad (26)$$

While this is certainly true for continuous operators, it does not hold for discrete differential operators. This leads to an error in the approximation of the numerical divergence of the new velocity. Thus, mass conservation is violated at the local level. Unlike the truncation error due to the order of spatial approximation, this error does not converge to zero with a decreasing spatial discretization.

In the context of Finite Volume Methods, this problem has been known for several decades. One proposed solution was to replace the classical discrete Laplace operator with wider stencils by taking the convolution of the divergence operator and the gradient operators, i.e. setting  $\Delta := \nabla \cdot \nabla$  at the discrete level [3]. Another, more widely used approach to overcome this problem is to use staggered grids in which the pressure and velocity fields are defined at different locations [15]. The staggered grid approach can not be generalized to meshfree methods since all properties, both scalars or vectors, are prescribed on the same nodes. Using the first approach of setting the numerical Laplace operator to be equal to the numerical divergence of the numerical gradient causes several issues in the meshfree context. Firstly, this would lead to different support sizes for the first and second derivatives. The second derivatives would be defined on a support double the size of that of the first derivative, which more than doubles the number of points in each support domain. Moreover, there is no control over the center stencil values, which makes diagonal dominance hard to achieve. As a result, convergence of the large linear systems can be troublesome. Thus, this problem still

persists in meshfree projection methods. Not only for meshfree GFDMs, but also in various other meshfree approximation methods including SPH [9, 42].

Other operator splitting algorithms such as the PISO algorithm [18], SIMPLE algorithms [31] and their derivatives also rely on pressure-Poisson equations. The difference being that the Poisson equations are derived by applying the divergence operator to the momentum conservation equation. Their meshfree equivalents possess the same drawback of  $\Delta \neq \nabla \cdot \nabla$  at the discrete level, which leads to the same inaccuracies in the approximation of the mass conservation condition.

#### *Compressible boundary layer:*

When using the classical GFDM approach of Section 2.1 the pressure-Poisson equation Eq. (24) is solved at interior points with appropriate boundary conditions on the boundary points. The divergence of the velocity at boundary points depends on the pressure-Poisson equation at boundary points which is not solved at all. This results in the formation of a numerical boundary layer of compressible fluid, with non-zero divergence of velocity, during the simulation of incompressible fluids. This problem has been alleviated by Tiwari and Kuhnert [39] by solving a system where an over-determined system is solved at boundary points, which considers both the relevant boundary conditions and the pressure-Poisson equation, by using the modified framework of Section 2.2. This problem of numerical boundary layers is also avoided by several solvers that solve the pressure-Poisson system before the implicit velocity system [45]. More recently, alternate solutions to this same problem were also considered by Idelsohn and Oñate [16], but specific to free surface boundaries. However, the earlier issue of  $\Delta \neq \nabla \cdot \nabla$  at the discrete level is present even at the boundaries, and is a larger source of error than the compressible boundary layer.

#### *Poor accuracy at low Reynolds flow:*

Projection methods usually suffer from low accuracy, and often instability, for fluid flows at low Reynolds numbers. This problem is reduced, but not overcome, by the implicit nature of Eq. (22). As a result, projection methods do not provide good approximations for fluids with high viscosity such as molten glass [29].

### *3.2. Monolithic Methods*

Coupled velocity-pressure solvers usually have the advantage of being more stable, especially for larger time-step sizes, but often come at the disadvantage of ill-conditioned systems. To avoid the problem of ill-conditioned systems, Kuhnert [22] developed a penalty formulation based on the classical GFDM approach of Section 2.1. In two spatial dimensions, for  $\vec{v} = (u, v)$  and  $\vec{g} = (g_x, g_y)$ , it can be written in matrix form as

$$\begin{pmatrix} I - \frac{\Delta t}{\rho} \eta \mathbf{C}^\Delta & & \frac{\Delta t}{\rho} \mathbf{C}^x \\ & I - \frac{\Delta t}{\rho} \eta \mathbf{C}^\Delta & \frac{\Delta t}{\rho} \mathbf{C}^y \\ \mathbf{C}^x & \mathbf{C}^y & -A \frac{\Delta t}{\rho} \end{pmatrix} \begin{pmatrix} \vec{U}^{(n+1)} \\ \vec{V}^{(n+1)} \\ \vec{P}_{corr} \end{pmatrix} = \begin{pmatrix} \vec{U}^{(n)} - \frac{\Delta t}{\rho} \mathbf{C}^x \vec{P}^* + \Delta t \vec{G}_x \\ \vec{V}^{(n)} - \frac{\Delta t}{\rho} \mathbf{C}^y \vec{P}^* + \Delta t \vec{G}_y \\ 0 \end{pmatrix}, \quad (27)$$

where  $\mathbf{C}^x$  is the matrix formed by the stencil coefficients for the numerical differential operators for the  $x$  derivative,  $c_{ij}^x$  (see Eq. (7) and Eq. (8)). Similarly for  $\mathbf{C}^y$  and  $\mathbf{C}^\Delta$ . The vector  $\vec{U}^{(n+1)}$  is formed by  $u^{(n+1)}$  at all points in the computational domain,  $\vec{U}^{(n+1)} = (u_1^{(n+1)}, \dots, u_N^{(n+1)})^T$  and similarly for the other upper case vectors. Thus, the first two blocks of rows in Eq. (27) represents the momentum conservation equation at all points. The last block of rows in Eq. (27) is a penalty formulation for the conservation of mass equation

$$\nabla \cdot \vec{v}^{(n+1)} - A \frac{\Delta t}{\rho} \Delta p_{corr} = 0. \quad (28)$$

Rows corresponding to boundary points in Eq. (27) are replaced with the appropriate problem specific boundary conditions. The ideal scenario would be to set  $A = 0$ , to get an exact conservation of mass. However, that leads to an ill-conditioned system. Further, typical iterative solvers do not converge for such a system due to a complete lack of diagonal dominance in the last block of rows in Eq. (27). While specialized

solvers for saddle-point problems [4, 26], could possibly be used to obtain solutions to these sparse linear systems, simulations are still usually unstable when  $A = 0$ . Thus, non-zero values of  $A$  need to be used. Kuhnert [22] takes this parameter to be in the range of  $A \in (0, 0.3)$ . Lower values of  $A$  are preferred for greater accuracy, however, higher values are needed for better conditioning and faster convergence of the resulting linear system. We note that using  $A = 1$  would result in an implicit and coupled projection method, with an additional velocity correction step, Eq. (23), required. The final pressure is given as done before  $p^{(n+1)} = p^* + p_{corr}$ .

This penalty approach has been shown to be more stable than the meshfree projection method [19, 22], especially at low Reynolds flows. However, setting  $A \neq 0$  in Eq. (28) leads to an artificial compressibility that is numerically observed to be similar to that introduced by the inconsistency between the  $\Delta$  and the  $\nabla \cdot \nabla$  discrete operators in operator splitting methods. Further, this approach has the same issue of a compressible boundary layer as that in classical operator splitting methods explained earlier.

This penalty approach coupled solver and the meshfree projection method have both been widely used and have shown to be robust methods with a wide variety of applications. Under the name of the Finite Pointset Method (FPM), they have also been used as the numerical basis of two commercially used meshfree simulation tools: NOGRID [29] and the meshfree module of VPS-PAMCRASH [40].

Other approaches to coupled solvers include solving the momentum conservation as in the first two blocks of rows in Eq. (27), with a pressure-Poisson equation replacing the third block of rows in Eq. (27). Like most fractional step methods, this solves the mass conservation indirectly, and introduces the same error as mentioned earlier of  $\Delta \neq \nabla \cdot \nabla$  at the discrete level. In mesh-based contexts, especially for Finite Element Methods, the equivalent systems of Eq. (27) with  $A = 0$  are often solved by algebraically decomposing the system which is essentially equivalent to a fractional step method [35]; or with the help of stability conditions [2] which introduce errors similar to that in the method described in this section.

#### 4. A New Monolithic Solver

We wish to solve the momentum and mass conservation equations as

$$\frac{\vec{v}^{(n+1)} - \vec{v}^{(n)}}{\Delta t} = \frac{\eta}{\rho} \Delta \vec{v}^{(n+1)} - \frac{1}{\rho} \nabla p^* - \frac{1}{\rho} \nabla p_{corr} + \vec{g}, \quad (29)$$

$$\nabla \cdot \vec{v}^{(n+1)} = 0. \quad (30)$$

We emphasize that the desire is to solve the mass conservation directly as in Eq. (30) and not indirectly via a pressure-Poisson equation. Using the classical GFDM approach of Section 2.1, Eq. (29) and Eq. (30) lead to Eq. (27) with  $A = 0$ . As mentioned earlier, this results in an ill-conditioned system that is very hard to solve with typical iterative procedures. This problem of ill-conditioned systems can be avoided by using Tiwari's approach presented in Section 2.2. For the same, we start by rewriting Eq. (29) and Eq. (30), in two spatial dimensions, to obtain

$$e_1 + u^{(n+1)} - \frac{\eta \Delta t}{\rho} \Delta u^{(n+1)} + \frac{\Delta t}{\rho} \partial^x p_{corr} = u^{(n)} - \frac{\Delta t}{\rho} \partial^x p^* + \Delta t g_x, \quad (31)$$

$$e_2 + v^{(n+1)} - \frac{\eta \Delta t}{\rho} \Delta v^{(n+1)} + \frac{\Delta t}{\rho} \partial^y p_{corr} = v^{(n)} - \frac{\Delta t}{\rho} \partial^y p^* + \Delta t g_y, \quad (32)$$

$$e_3 + \nabla \cdot \vec{v}^{(n+1)} = 0, \quad (33)$$

where  $e_1$ ,  $e_2$  and  $e_3$  are the errors in the discretizations of the respective PDEs. A quadratic minimization of  $e_1$ ,  $e_2$  and  $e_3$ , along with the errors from the Taylor expansions in all three variables,  $u$ ,  $v$  and  $p_{corr}$  can be done, similar to that done in Section 2.2. This results in a system of equations similar to Eq. (17) and provides better conditioned systems than Eq. (27) with  $A = 0$ . However, resulting simulations are mostly unstable. Thus, such an approach of using Tiwari's framework of GFDM discretization to coupled solvers has not been used successfully in the past.



A possible explanation of the instability is the lack of information about the pressure correction  $p_{corr}$ . To correct this and to introduce a further coupling condition between the velocity and pressure, we make use of the fact that this modified GFDM framework can be used to solve algebraically over-determined problems. The system of Eq. (31) – Eq. (33) are augmented with a pressure-Poisson equation to form an over-determined system. The Poisson equation is obtained in a manner similar to that done by SIMPLE and PISO methods, by applying the divergence operator to the conservation of momentum equation Eq. (20) to obtain

$$e_4 + \Delta p_{corr} + \rho \nabla \cdot \left[ \left( \vec{v}^{(n)} \cdot \nabla \right) \vec{v}^{(n+1)} \right] = \frac{\rho}{\Delta t} \nabla \cdot \vec{v}^{(n)} - \Delta p^* + \rho \nabla \cdot \vec{g}. \quad (34)$$

Eq. (31) – Eq. (34) are solved at all interior points. Note that the mass conservation condition is solved directly in Eq. (33) and indirectly in Eq. (34). At boundary points, the mass conservation condition Eq. (33) is solved in addition to the relevant boundary conditions, once again leading to more PDEs than variables. The errors in each of these PDEs or boundary conditions are minimized simultaneously with the errors in the Taylor expansions for each velocity component and the pressure.

$$\min J = \sum_{j \in S_i} W_j (e_j^u)^2 + \sum_{j \in S_i} W_j (e_j^v)^2 + \sum_{j \in S_i} W_j (e_j^p)^2 + \sum_{k=1}^4 W_{PDE,k} (e_k)^2, \quad (35)$$

where  $e_j^u$ ,  $e_j^v$  and  $e_j^p$  are the errors in the Taylor expansions around point  $i$  of  $u$ ,  $v$  and  $p_{corr}$  respectively. For all simulations,  $W_j$  are taken to as in Eq. (4), and  $W_{PDE,k} = 2$ . Thus, the following least squares system is solved at each interior point to obtain the function approximation stencil coefficients :  $\vec{E} = M\vec{a} - \vec{b}$  with

$$\vec{E} = (e_1^u, \dots, e_n^u, e_1^v, \dots, e_n^v, e_1^p, \dots, e_n^p, e_1, \dots, e_4)^T, \quad (36)$$

$$\vec{a} = (u, u_x, u_y, u_{xx}, u_{yy}, u_{xy}, v, v_x, v_y, v_{xx}, v_{yy}, v_{xy}, p, p_x, p_y, p_{xx}, p_{yy}, p_{xy})^T, \quad (37)$$

$$\vec{b} = (u_1, \dots, u_n, v_1, \dots, v_n, p_1, \dots, p_n, r_1, \dots, r_4)^T, \quad (38)$$

where  $p$  is used as shorthand for  $p_{corr}$ , and  $r_k$  are the right hand sides of equations Eq. (31) – Eq. (34). Further,

$$M = \begin{pmatrix} M_T & & \\ & M_T & \\ & & M_T \\ M_{P1} & M_{P2} & M_{P3} \end{pmatrix}, \quad (39)$$

with  $M_T$  being the parts coming from the Taylor expansions

$$M_T = \begin{pmatrix} 1 & \delta x_1 & \delta y_1 & \frac{1}{2}\delta x_1^2 & \frac{1}{2}\delta y_1^2 & \delta x_1 \delta y_1 \\ & & \vdots & \vdots & & \\ 1 & \delta x_n & \delta y_n & \frac{1}{2}\delta x_n^2 & \frac{1}{2}\delta y_n^2 & \delta x_n \delta y_n \end{pmatrix}, \quad (40)$$

and the part coming from the PDEs is given by

$$\left( \begin{array}{ccc|cccccc|cccc|cccc} M_{P1} & M_{P2} & M_{P3} & & & & & & & & & & & & & & & & & \end{array} \right) = \left( \begin{array}{cccc|cccc|cccc|cccc} 1 & 0 & 0 & -\frac{\eta \Delta t}{\rho} & -\frac{\eta \Delta t}{\rho} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\Delta t}{\rho} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -\frac{\eta \Delta t}{\rho} & -\frac{\eta \Delta t}{\rho} & 0 & 0 & 0 & \frac{\Delta t}{\rho} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \rho u_x^{(n)} & \rho v_x^{(n)} & 0 & 0 & 0 & 0 & \rho u_y^{(n)} & \rho v_y^{(n)} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{array} \right). \quad (41)$$

The first two rows in Eq. (41) represent the momentum conservation in  $x$  and  $y$  directions respectively, the third row represents the mass conservation equation, and the last row represents the pressure-Poisson equation. Similar systems are obtained for boundary points, with the relevant rows in Eq. (41) replaced

with the boundary conditions, as done in [39]. As done earlier, a formal minimization leads to  $\vec{a} = [(M^T W M)^{-1} M^T W] \vec{b}$ . Only the function approximation stencils are of interest to us, i.e. the rows of  $u$ ,  $v$  and  $p_{corr}$ . They can be written as

$$u_i = \sum_{j \in S_i} \alpha_{ij}^u u_j + \sum_{j \in S_i} \beta_{ij}^u v_j + \sum_{j \in S_i} \gamma_{ij}^u (p_{corr})_j + \sum_{k=1}^4 \zeta_{ik}^u r_k, \quad (42)$$

$$v_i = \sum_{j \in S_i} \alpha_{ij}^v u_j + \sum_{j \in S_i} \beta_{ij}^v v_j + \sum_{j \in S_i} \gamma_{ij}^v (p_{corr})_j + \sum_{k=1}^4 \zeta_{ik}^v r_k, \quad (43)$$

$$(p_{corr})_i = \sum_{j \in S_i} \alpha_{ij}^p u_j + \sum_{j \in S_i} \beta_{ij}^p v_j + \sum_{j \in S_i} \gamma_{ij}^p (p_{corr})_j + \sum_{k=1}^4 \zeta_{ik}^p r_k, \quad (44)$$

for  $i = 1, \dots, N$ . The coefficients  $\alpha$ ,  $\beta$ ,  $\gamma$  and  $\zeta$  represent the values in the relevant row of the matrix  $[(M^T W M)^{-1} M^T W]$ . These can be rearranged to obtain the sparse linear system

$$(1 - \alpha_{ii}^u) u_i - \sum_{\substack{j \in S_i \\ j \neq i}} \alpha_{ij}^u u_j - \sum_{j \in S_i} \beta_{ij}^u v_j - \sum_{j \in S_i} \gamma_{ij}^u (p_{corr})_j = \sum_{k=1}^4 \zeta_{ik}^u r_k, \quad (45)$$

$$- \sum_{j \in S_i} \alpha_{ij}^v u_j + (1 - \beta_{ii}^v) v_i - \sum_{\substack{j \in S_i \\ j \neq i}} \beta_{ij}^v v_j - \sum_{j \in S_i} \gamma_{ij}^v (p_{corr})_j = \sum_{k=1}^4 \zeta_{ik}^v r_k, \quad (46)$$

$$- \sum_{j \in S_i} \alpha_{ij}^p u_j - \sum_{j \in S_i} \beta_{ij}^p v_j + (1 - \gamma_{ii}^p) (p_{corr})_i - \sum_{\substack{j \in S_i \\ j \neq i}} \gamma_{ij}^p (p_{corr})_j = \sum_{k=1}^4 \zeta_{ik}^p r_k. \quad (47)$$

This resulting sparse linear system is solved using an iterative solver. Eq. (45) – Eq. (47) produce a diagonally dominant system. In Eq. (45), for example, the magnitude of the diagonal values of  $1 - \alpha_{ii}^u$  are significantly larger than that of the off-diagonal values of  $\alpha_{ij}^u$   $j \neq i$ ,  $\beta_{ij}^u$  and  $\gamma_{ij}^u$ . Thus, the resulting linear system converges well with typical iterative procedures.

A short comparison between the meshfree projection method presented in Section 3.1, the penalty approach coupled solver of Section 3.2 and the new coupled solver presented in this section is listed below

- All three methods have the same theoretical convergence rate with the spatial discretization as all use Taylor expansions up to the same order of accuracy. All three methods also use similar first order temporal discretizations. Higher order methods have been studied extensively, especially for mesh-based fractional step methods (for example, [43]). Such higher order approximations could be applied to all three methods considered here.
- All three methods are “approximate” methods as apposed to “exact” methods in the sense that they only solve the mass conservation equation up to the truncation error. However, the new method does not contain the additional sources of error present in the other two methods. The projection method solves the mass conservation indirectly which leads to errors due to a lack of consistency between the first and second order derivatives. The penalty approach coupled solver attempts to solve the mass conservation directly, but introduces an artificial compressibility to improve conditioning of the resulting system. On the other hand, the new coupled solver solves the mass conservation equation directly and without introducing an artificial compressibility, and thus provides a much better approximation to the mass conservation equation than both other methods.
- While the penalty approach coupled solver results in a compressible boundary layer, this is not present in the projection method of Tiwari and Kuhnert [39] or the new coupled solver presented here, due to the addition of the mass balance equation on boundary points.

- Numerically it is observed that the new coupled solver has stability comparable to the penalty approach coupled solver, and thus, much better than the projection method.
- Both coupled solvers solve one large implicit linear system while the projection method solves two smaller implicit linear systems. In both coupled solvers, the linear systems are of the same size, but the system is denser in the new coupled solver. The resulting system of Eq. (45) – Eq. (47) can be written in matrix form as

$$\begin{pmatrix} I - \alpha^u & -\beta^u & -\gamma^u \\ -\alpha^v & I - \beta^v & -\gamma^v \\ -\alpha^p & -\beta^p & I - \gamma^p \end{pmatrix} \begin{pmatrix} \vec{U}^{(n+1)} \\ \vec{V}^{(n+1)} \\ \vec{P}_{corr} \end{pmatrix} = \begin{pmatrix} \vec{R}_1 \\ \vec{R}_2 \\ \vec{R}_3 \end{pmatrix}, \quad (48)$$

where  $\alpha$ ,  $\beta$  and  $\gamma$  are matrices formed from the coefficients in Eq. (45) – Eq. (47) and  $\vec{R}_1$ ,  $\vec{R}_2$  and  $\vec{R}_3$  are vectors formed from the right hand sides of Eq. (45) – Eq. (47) respectively. While the sparsity pattern of each block of rows in Eq. (48) and Eq. (27) are identical, as they are dependent only on the support domains for each point, several zero blocks are present in Eq. (27), while all blocks are non-zero in Eq. (48). This is no longer true for cases of spatially varying viscosity  $\eta$ , which are very common in fluid flow applications, for which all blocks are non-zero in the systems arising from both the coupled solvers.

- The larger size of the implicit linear systems means that both coupled solvers have higher memory requirements than the projection method.
- The fastest simulation times between the three methods vary on a case by case basis, but the overall simulation time is similar for all three methods.

## 5. Numerical Results

The explicit time-integration for the movement of points according to Eq. (21) results in a CFL-like condition on the time-step size

$$\Delta t = C_{\Delta t} \left( \frac{h}{\|\vec{v}\|_{min}} \right). \quad (49)$$

Thus, a varying time-step size according to Eq. (49) is used in all performed simulations. To determine the numerical order of accuracy of the three methods, a small time-step is used in the first test case below by using a small value of  $C_{\Delta t}$ . The remaining test cases use a much larger time-step which results in lower experimental convergence rates. In the comparison of simulation times, the total time of the simulation refers to the total clock time (in seconds) of the simulation, including the initial point cloud setup and the post-processing error calculations. To ensure that these comparisons are realistic, all three methods were implemented by the same programmer and use identical memory management. All simulations were carried out in Fortran and were run serially on an Intel XeonE5-2670 CPU rated at 2.60GHz. All sparse linear systems are solved using the BiCGSTAB iterative solver [41] without the use of any preconditioner. For the penalty approach coupled solver, the penalty coefficient is taken to be  $A = 0.1$  in all simulations, since lower values result in much larger simulation times due to poor convergence of the linear systems. In all the figures below, the projection method presented in Section 3.1 is referred to by ‘Projection’, the coupled solver with the penalty formulation presented in Section 3.2 is referred to by ‘Coupled: Penalty’ and the new coupled solver which directly solves an algebraically over-determined system, as done in Section 4, is referred to as ‘Coupled: New’.

### 5.1. Taylor-Green Vortices

As a validation case, we consider the two-dimensional decaying vortices referred to as the Taylor-Green vortices on  $[0, 2\pi] \times [0, 2\pi]$ . The analytical solution is given by

$$u_a = \sin(x) \cos(y) \exp(-4\pi t/Re), \quad (50)$$

$$v_a = -\cos(x) \sin(y) \exp(-4\pi t/Re), \quad (51)$$

$$p_a = \frac{\rho}{4}(\cos(2x) + \sin(2y)) \exp(-8\pi t/Re), \quad (52)$$

$$\vec{g} = 0, \quad (53)$$

where  $\vec{v}_a = (u_a, v_a)$  is the analytical solution for the velocity,  $p_a$  is the analytical solution for the pressure, and  $Re = \frac{\rho UL}{\eta}$  is the Reynolds number, with the characteristic velocity  $U = 1$  and the characteristic length  $L = 2\pi$ . Error in the numerical solution  $\vec{v}$  is measured by

$$\epsilon_2 = \left[ \frac{\sum_{i=1}^N \|\vec{v}_i - \vec{v}_a(\vec{x}_i)\|^2 V_i}{\sum_{i=1}^N \|\vec{v}_a(\vec{x}_i)\|^2 V_i} \right]^{\frac{1}{2}}, \quad (54)$$

where  $V_i$  is a volume associated with point  $i$ . For an error  $\epsilon$  and consecutive smoothing lengths  $h_1$  and  $h_2$ , the rate of convergence of the solution with changing smoothing length is measured as

$$r = \frac{\log\left(\frac{\epsilon(h_2)}{\epsilon(h_1)}\right)}{\log\left(\frac{h_2}{h_1}\right)}. \quad (55)$$

The initial condition for the velocity is taken in accordance with the exact solution, and is shown in Figure 1. Dirichlet boundary conditions are used on all boundaries. The simulations are done up to an ending time of  $t_{end} = 1s$ , for  $\eta = 1Pa s$  and  $\rho = 1kg/m^3$ , which gives  $Re = 2\pi$ . A small time-step is used according to Eq. (49) with  $C_{\Delta t} = 0.005$ . The convergence of the errors with the smoothing length  $h$  are shown in Figure 2 and are tabulated in Table 1. All three methods match the analytical solution well. All three methods exhibit a similar convergence rate, which matches the theoretical expectation. Errors in the new coupled solver are smaller than the other two methods, while the errors in the projection method are the largest, but are only slightly larger than those in the penalty approach coupled solver. Total simulation times for each case are also shown in Table 1. All three methods take approximately the same time, with the new coupled solver being the slowest and the projection method being the fastest. Similar results were obtained for larger Reynolds numbers.

Table 1: Errors, convergence orders and simulation times for the Taylor-Green vortices test case.  $h$  is the smoothing length,  $N$  is the number of points in the entire domain at the initial state,  $\epsilon_2$  is the relative error,  $r$  is the order of convergence of  $\epsilon_2$ , and  $t$  is the simulation time in seconds.

$h$	$N$	Projection			Coupled:Penalty			Coupled:New		
		$\epsilon_2$	$r$	$t$	$\epsilon_2$	$r$	$t$	$\epsilon_2$	$r$	$t$
1.0	293	$3.1 \times 10^{-2}$	—	6	$2.4 \times 10^{-2}$	—	7	$1.2 \times 10^{-2}$	—	9
0.5	1 047	$9.7 \times 10^{-3}$	1.67	28	$6.5 \times 10^{-3}$	1.88	29	$3.1 \times 10^{-3}$	1.95	32
0.25	3 856	$3.2 \times 10^{-3}$	1.59	195	$2.5 \times 10^{-3}$	1.37	202	$1.0 \times 10^{-3}$	1.63	207
0.125	14 878	$1.1 \times 10^{-3}$	1.54	1934	$8.2 \times 10^{-4}$	1.60	2010	$3.3 \times 10^{-4}$	1.59	2031

### 5.2. Bifurcated Tube

We consider flow of a fluid through the bifurcated tube shown in Figure 3. The length of the tube is  $60m$ , the width is  $4m$  in the thick region and  $1m$  in the thin region. Simulation parameters are set as  $t_{end} = 1s$ ,  $\rho = 10^3kg/m^3$  and  $\eta = 2Pa s$ . The velocity at the inflow, on the left of the tube, is kept

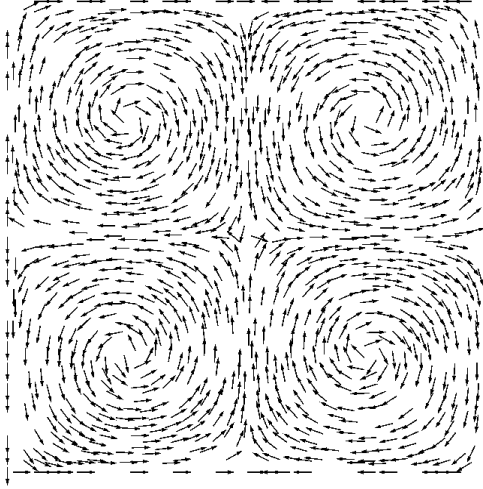


Figure 1: Initial condition for Taylor Green vortices. Arrow lengths are constant and are not scaled by velocity magnitudes.

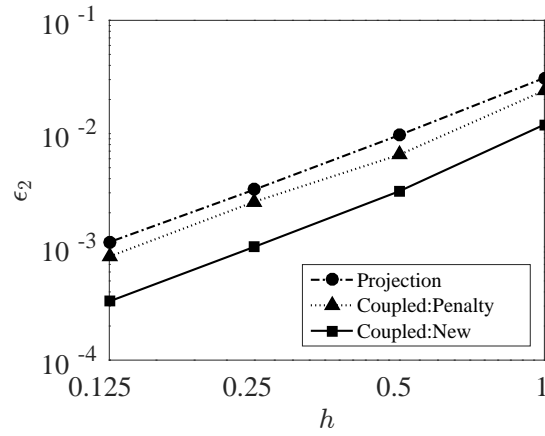


Figure 2: Convergence of error for Taylor-Green vortices.

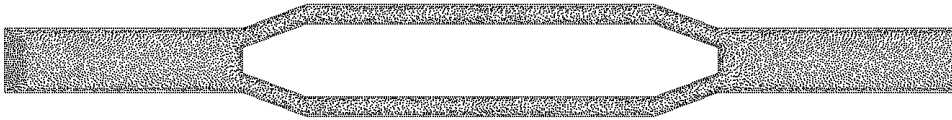


Figure 3: 2D Bifurcated tube. Fluid inflow is on the left, and outflow is on the right.

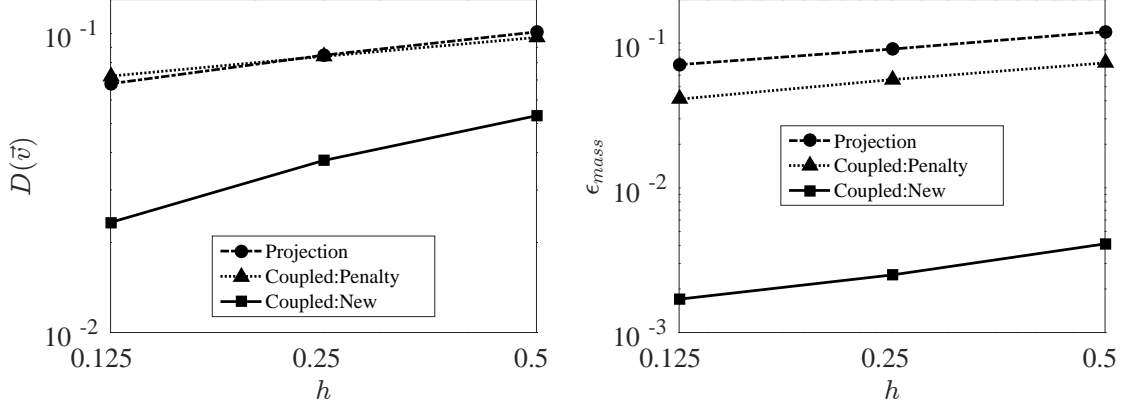


Figure 4: Bifurcated tube: average divergence of velocity in the simulation domain (left) and error in mass conservation (right).

constant at  $\vec{v}_{in} = (2m/s, 0)$ . This results in a Reynolds number of the order of  $10^3$ . A varying time-step is used according to Eq. (49) with  $C_{\Delta t} = 0.05$ . Homogeneous Neumann boundary conditions for the velocity are used at the outflow and no-slip conditions on the walls. The pressure is kept constant at atmospheric pressure at the outflow and homogeneous Neumann boundary conditions are considered elsewhere for the pressure. The error in mass conservation is measured as the difference between the total volume of fluid flowing in and that flowing out, throughout the entire simulation. The mass conservation error is given by

$$\epsilon_{mass} = \left| \frac{\int_0^{t_{end}} \left[ \int_{\partial\Omega_{in}} \vec{n} \cdot \vec{v} dA \right] dt + \int_0^{t_{end}} \left[ \int_{\partial\Omega_{out}} \vec{n} \cdot \vec{v} dA \right] dt}{\int_0^{t_{end}} \left[ \int_{\partial\Omega_{in}} \vec{n} \cdot \vec{v} dA \right] dt} \right|. \quad (56)$$

where  $\vec{n}$  is the outward pointing unit normal and  $\partial\Omega_{in}$  and  $\partial\Omega_{out}$  are the inflow and outflow boundaries respectively. Note that  $\epsilon_{mass}$  measures the errors during transient states too, and not just the errors in the steady state solution. A measure for the velocity divergence throughout the domain is taken as the integral of the divergence of velocity scaled by the total volume

$$D(\vec{v}) = \frac{\int_{\Omega} |\nabla \cdot \vec{v}| dV}{\int_{\Omega} dV}. \quad (57)$$

This can be interpreted as the average value of local error in the mass conservation equation. Note that there is no direct correlation between the measures of divergence and mass conservation,  $D(\vec{v})$  and  $\epsilon_{mass}$ , because the absolute value of velocity divergence is taken in  $D(\vec{v})$ . The presence of a numerical source and a sink of equal magnitudes for  $\nabla \cdot \vec{v}$  would cancel out while measuring the mass conservation, but they would add up while measuring  $D(\vec{v})$ . Figure 4 shows the convergence of the velocity divergence averaged over all time-steps, and the convergence of the error in mass conservation with respect to changing smoothing length  $h$ . The same are also tabulated in Table 2. The errors follow a similar pattern to those in the Taylor-Green vortices test case. The new coupled solver produces significantly smaller errors than the other two methods in both the average velocity divergence and mass conservation. The difference between the projection method and the coupled penalty approach is quite small. We note that convergence orders are observed to be much smaller than in the earlier test case due to the use of larger time-steps, Neumann boundary conditions and a non-standard domain geometry.

Simulation times for the three methods are also shown in Table 3. While the new coupled solver is the slowest by a large margin on the coarsest point cloud, it is the fastest on the finest point cloud. This could be due to slower convergence of linear systems in the other two methods.

To illustrate that the difference between the new coupled solver presented in this paper and the penalty formulation coupled solver goes beyond the addition of the zero divergence equation at the boundary points,

Table 2: Errors, convergence orders and simulation times for the bifurcated tube test case.  $h$  is the smoothing length,  $N$  is the number of points in the entire domain at the initial state,  $\epsilon_{mass}$  is the relative error in mass conservation,  $r$  is the order of convergence of  $\epsilon_{mass}$ , and  $t$  is the simulation time in seconds.

$h$	$N$	Projection			Coupled:Penalty			Coupled:New		
		$\epsilon_{mass}$	$r$	$t$	$\epsilon_{mass}$	$r$	$t$	$\epsilon_{mass}$	$r$	$t$
0.5	5 805	$1.2 \times 10^{-1}$	—	556	$7.3 \times 10^{-2}$	—	718	$4.1 \times 10^{-3}$	—	927
0.25	21 294	$9.1 \times 10^{-2}$	0.40	7378	$5.6 \times 10^{-2}$	0.38	8752	$2.5 \times 10^{-3}$	0.71	6056
0.125	81 125	$7.1 \times 10^{-2}$	0.36	76850	$4.1 \times 10^{-2}$	0.45	76936	$1.7 \times 10^{-3}$	0.56	53477

we split the average divergence of velocity, Eq. (57), along the interior and boundary points  $D(\vec{v}) = D_{int}(\vec{v}) + D_{bnd}(\vec{v})$ , with

$$D_{int}(\vec{v}) = \frac{\int_{\Omega \setminus \partial\Omega} |\nabla \cdot \vec{v}| dV}{\int_{\Omega} dV}; \quad D_{bnd}(\vec{v}) = \frac{\int_{\partial\Omega} |\nabla \cdot \vec{v}| dV}{\int_{\Omega} dV}, \quad (58)$$

where  $\Omega \setminus \partial\Omega$  represents only the interior points. Figure 5 shows  $D_{int}(\vec{v})$  and  $D_{bnd}(\vec{v})$  averaged over all time-steps. It illustrates that the new coupled solver improves the accuracy of the mass conservation condition across both interior and boundary points.

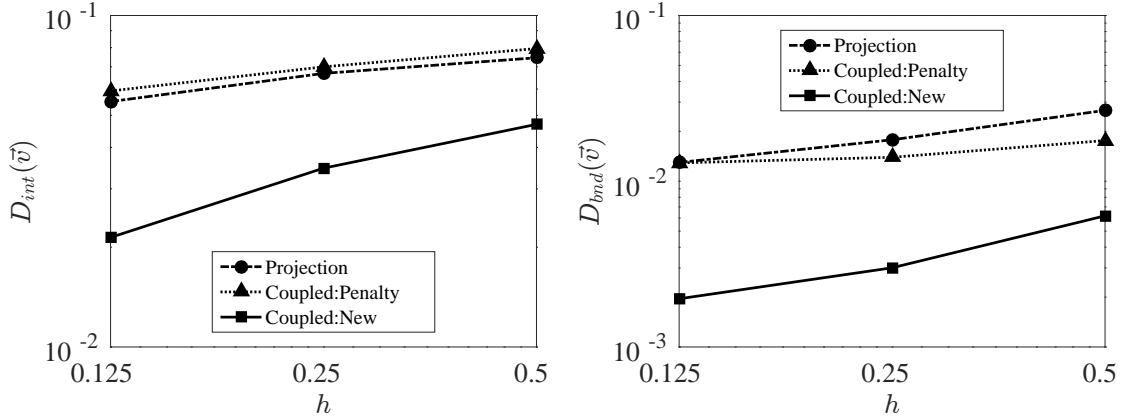


Figure 5: Velocity divergence on interior points (left) and boundary points (right) for the bifurcated tube test case.

### 5.3. Sloshing

The most common area of application of Lagrangian meshfree methods is for flows with moving free surfaces. We consider the sloshing of a fluid contained in a constantly moving rectangular box as shown in Figure 6. The dimensions of the initial state of the fluid are  $1.2m \times 0.12m$ , and that of the box containing the fluid are  $1.2m \times 0.6m$ .

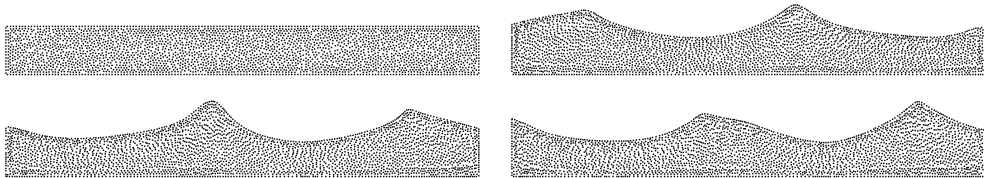


Figure 6: Sloshing at times  $t = 0s$  (top left),  $t = 1.31s$  (top right),  $t = 1.63s$  (bottom left), and  $t = 2.23s$  (bottom right).

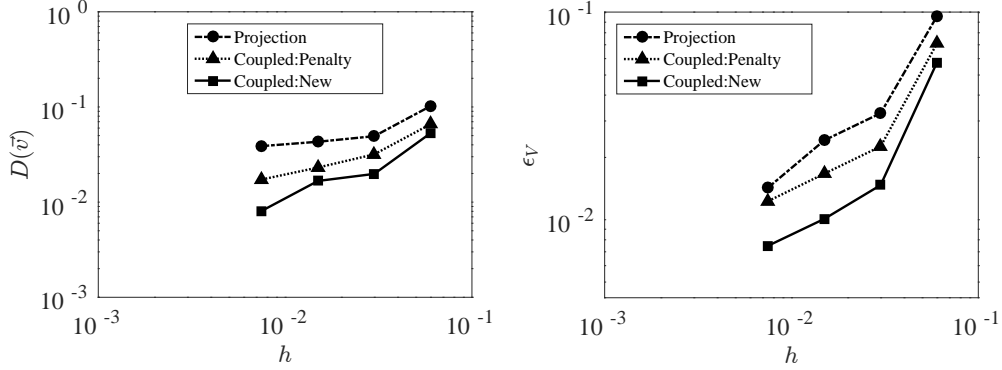


Figure 7: Sloshing: convergence of solution with smoothing length  $h$ . Divergence of velocity field (left) and error in mass conservation (right).

The initial state is taken to be at rest. Slip boundary conditions are used at the walls for the velocity. The free surface boundary conditions are given by

$$\begin{aligned}\vec{t}^T \cdot \mathbf{S} \cdot \vec{n} &= 0 \\ \vec{n}^T \cdot \mathbf{S} \cdot \vec{n} &= p - p_0\end{aligned}$$

where  $\vec{t}$  is a unit vector in the tangential direction to the free surface,  $p_0$  is the atmospheric pressure, and  $\mathbf{S} = \mathbf{S}(\vec{v})$  is the stress tensor. Homogeneous Neumann boundary conditions are used for the pressure at the walls. The movement of the box is represented in the gravitational and body forces term by setting  $\vec{g} = (2 \cos(10t), -10)$ . The simulation parameters are set as  $t_{end} = 3s$ ,  $\rho = 10^3 kg/m^3$ , and  $\eta = 0.1 Pa s$ , which leads to a Reynolds number of the order of  $10^4$ . A varying time-step is used according to Eq. (49) with  $C_{\Delta t} = 0.3$ . The error in mass conservation is measured by the change in total volume occupied by all points, since the density  $\rho$  is fixed and constant throughout the domain

$$\epsilon_V = \frac{|\int_{\Omega_0} dV - \int_{\Omega_{end}} dV|}{\int_{\Omega_0} dV}, \quad (59)$$

where  $\Omega_0$  is the initial domain and  $\Omega_{end}$  is the domain at  $t_{end}$ . The average velocity divergence is measured as done earlier in Eq. (57). The convergence of volume conservation error and average velocity divergence  $D(\vec{v})$  with respect to a changing smoothing length  $h$  for all three methods are shown in Figure 7 and are tabulated in Table 3. The results follow a similar trend to that in the earlier two test cases. The new coupled solver shows the highest accuracy in both velocity divergence and mass conservation while the projection method shows the least accuracy in both. However, the difference between the new method and the two older methods is not as large as in the previous test case. The sharp increase in accuracy between  $h = 0.06$  and  $h = 0.03$  is due to a bad approximation of the free surface at the coarsest grid. For the finer point clouds, a small convergence rate is observed once again due to the boundary conditions used and the use of large time-steps.

Simulation times for the three methods are also present in Table 3. All three methods take almost the same simulation time, with the new coupled solver being the slowest for 3 out of the 4 point clouds considered.

We note that more turbulent sloshing problems than the one considered here produce larger errors in volume conservation, but for such problems, the largest source of error is in the management of the point cloud and not in the approximation of the PDEs.

#### 5.4. Accuracy of Gradient Reconstruction

In the classical meshfree GFDM approach of Section 2.1, the errors in Taylor expansions are minimized directly. As a result, first order monomials are differentiated exactly, which is evident from the formulation



Table 3: Errors, convergence orders and simulation times for the sloshing test case.  $h$  is the smoothing length,  $N$  is the number of points in the entire domain at the initial state,  $\epsilon_V$  is the error in volume conservation,  $r$  is the order of convergence of  $\epsilon_V$ , and  $t$  is the simulation time in seconds.

$h$	$N$	Projection			Coupled:Penalty			Coupled:New		
		$\epsilon_V$	$r$	$t$	$\epsilon_V$	$r$	$t$	$\epsilon_V$	$r$	$t$
0.06	382	$9.59 \times 10^{-2}$	—	10	$7.09 \times 10^{-2}$	—	12	$5.73 \times 10^{-2}$	—	15
0.03	1 361	$3.27 \times 10^{-2}$	1.56	51	$2.26 \times 10^{-2}$	1.65	59	$1.47 \times 10^{-2}$	1.96	68
0.015	5 119	$2.42 \times 10^{-2}$	0.43	723	$1.67 \times 10^{-2}$	0.44	602	$1.01 \times 10^{-2}$	0.54	746
0.0075	19 704	$1.43 \times 10^{-2}$	0.76	7871	$1.23 \times 10^{-2}$	0.44	6685	$0.75 \times 10^{-2}$	0.43	7268

in Eq. (12). However, the same need not be true for the modified approach of Section 2.2 used for the new coupled solver presented in this paper. The addition of the PDE error terms to the functional minimization in Eq. (15) and Eq. (35) could result in larger errors in the Taylor expansions. This would introduce errors in gradient reconstruction.

For  $\vec{v} = (u, v)$ , we examine this difference numerically by looking at the errors in the Taylor expansions for  $u$ . Formally, the functional being compared is  $J = \sum_{j \in S_i} W_j (e_j^u)^2$  for each point  $i$ . After the velocity at the new time-level is computed, the obtained velocity is checked for errors in the Taylor expansions. Thus, these errors include not just the truncation error in the discretization, but also the error due to the tolerance of the sparse iterative solver. This comparison is done using the Taylor-Green vortices test case considered earlier in Section 5.1, and for velocities after the completion of the first time step of the simulation. These errors are tabulated in Table 4 for the classical GFDM (with the projection method) and the modified framework used for the new coupled solver. The errors are larger in the new coupled solver, but not significantly.

Table 4: Errors in Taylor expansions measured by the functional  $J = \sum_{j \in S_i} W_j (e_j^u)^2$ . The table shows the mean value of  $J$  across all interior points at the end of the first time-step for the Taylor-Green vortices test case.

$h$	Classical GFDM: Projection	Modified GFDM: New Coupled Solver
1.0	0.2976	0.3057
0.5	0.09579	0.09709
0.25	0.03255	0.03345
0.125	0.01153	0.01470

## 6. Conclusion

We presented a new monolithic algorithm for the incompressible Navier–Stokes equations, solved using a meshfree Generalized Finite Difference Method (GFDM). While existing algorithms either solve the mass conservation indirectly via a pressure-Poisson equation or introduce an artificial compressibility, in the new method presented here, the mass conservation is solved directly without the introduction of any artificial compressibility. This results in improved local approximations for the mass conservation equation for both interior and boundary points, which results in better accuracy overall. In this method, the momentum and mass conservation equations are solved together, and simultaneously with a pressure-Poisson equation. The addition of this Poisson equation is essential for stabilizing the scheme and results in an over-determined system of PDEs. Accuracy is further improved at boundary points by solving the mass conservation equation in addition to the usual boundary conditions.

This new scheme was compared with two existing methods: one fractional step method and one monolithic method. All three methods have the same spatial and temporal order of accuracy. Numerical comparisons were done for different Reynolds flows, and the new method was shown to produce more accurate results.

Our future work in this direction will be devoted to extending this algorithm to three spatial dimensions. A further interesting point of study is the impact of the ratio of weights in the minimization of the functional in Eq. (35). Higher weights could be used, for example, for the minimization of the error in the mass conservation equation.

## References

- [1] A. S. Almgren, J. B. Bell, and W. G. Szymczak. A numerical method for the incompressible navier-stokes equations based on an approximate projection. *SIAM Journal on Scientific Computing*, 17(2):358–369, 1996.
- [2] A. R. E. Antunes, P. R. M. Lyra, R. B. Willmersdorf, and S. M. A. Bastos. An implicit monolithic formulation based on finite element formulation for incompressible navier–stokes equations. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 37(1):199–210, 2015.
- [3] J. B. Bell, P. Colella, and L. H. Howell. An efficient second-order projection method for viscous incompressible flow. In *Proceedings of the Tenth AIAA Computational Fluid Dynamics Conference, AIAA*, volume 360, 1991.
- [4] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14:1–137, 2005.
- [5] H. Bhatia, G. Norgard, V. Pascucci, and P.-T. Bremer. The helmholtz-hodge decomposition—a survey. *IEEE Transactions on Visualization and Computer Graphics*, 19(8):1386–1404, Aug. 2013.
- [6] B. Boroomand, A. A. Tabatabaei, and E. Oñate. Simple modifications for stabilization of the finite point method. *International Journal for Numerical Methods in Engineering*, 63(3):351–379, 2005.
- [7] D. L. Brown, R. Cortez, and M. L. Minion. Accurate projection methods for the incompressible Navier–Stokes equations. *Journal of Computational Physics*, 168(2):464–499, 2001.
- [8] A. J. Chorin. Numerical solution of the navier-stokes equations. *Mathematics of computation*, 22(104):745–762, 1968.
- [9] S. J. Cummins and M. Rudman. An sph projection method. *Journal of Computational Physics*, 152(2):584 – 607, 1999.
- [10] C. Drumm, S. Tiwari, J. Kuhnert, and H.-J. Bart. Finite pointset method for simulation of the liquid - liquid flow field in an extractor. *Computers & Chemical Engineering*, 32(12):2946 – 2957, 2008.
- [11] J. Fang and A. Parriaux. A regularized lagrangian finite point method for the simulation of incompressible viscous flows. *Journal of Computational Physics*, 227(20):8894 – 8908, 2008.
- [12] A. Fedoseyev, M. Friedman, and E. Kansa. Improved multiquadric method for elliptic partial differential equations via pde collocation on the boundary. *Computers & Mathematics with Applications*, 43(3):439 – 455, 2002.
- [13] L. Gavete, M. Gavete, and J. Benito. Improvements of generalized finite difference method and comparison with other meshless method. *Applied Mathematical Modelling*, 27(10):831 – 847, 2003.
- [14] A. Ghosh and S. Deshpande. Least squares kinetic upwind method for inviscid compressible flows. *AIAA paper*, 1735:1995, 1995.
- [15] F. H. Harlow and J. E. Welch. Numerical calculation of timedependent viscous incompressible flow of fluid with free surface. *Physics of Fluids*, 8(12):2182–2189, 1965.
- [16] S. R. Idelsohn and E. Oñate. The challenge of mass conservation in the solution of free-surface flows with the fractional-step method: Problems and solutions. *International Journal for Numerical Methods in Biomedical Engineering*, 26(10):1313–1330, 2010.
- [17] O. Iliev and S. Tiwari. A generalized (meshfree) finite difference discretization for elliptic interface problems. In *Revised Papers from the 5th International Conference on Numerical Methods and Applications*, NMA '02, pages 488–497, London, UK, 2003. Springer-Verlag.
- [18] R. Issa. Solution of the implicitly discretised fluid flow equations by operator-splitting. *Journal of Computational Physics*, 62(1):40 – 65, 1986.
- [19] A. Jefferies, J. Kuhnert, L. Aschenbrenner, and U. Giffhorn. Finite pointset method for the simulation of a vehicle travelling through a body of water. In M. Griebel and A. M. Schweitzer, editors, *Meshfree Methods for Partial Differential Equations VII*, pages 205–221, Cham, 2015. Springer International Publishing.
- [20] A. Katz and A. Jameson. Meshless scheme based on alignment constraints. *AIAA journal*, 48(11):2501–2511, 2010.
- [21] J. Kuhnert. An upwind finite pointset method (FPM) for compressible euler and navier-stokes equations. In M. Griebel and M. A. Schweitzer, editors, *Meshfree Methods for Partial Differential Equations*, pages 239–249, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.
- [22] J. Kuhnert. Meshfree numerical scheme for time dependent problems in fluid and continuum mechanics. In S. Sundar, editor, *Advances in PDE Modeling and Computation*, pages 119–136. Anne Books, 2014.
- [23] T. Liszka and J. Orkisz. Special issue-computational methods in nonlinear mechanics the finite difference method at arbitrary irregular grids and its application in applied mechanics. *Computers and Structures*, 11(1):83 – 95, 1980.
- [24] G.-R. Liu and M. B. Liu. *Smoothed particle hydrodynamics: a meshfree particle method*. World Scientific, 2003.
- [25] F. Maria Denaro. On the application of the helmholtzhodge decomposition in projection methods for incompressible flows with general boundary conditions. *International Journal for Numerical Methods in Fluids*, 43(1):43–69, 2003.
- [26] B. Metsch. *Algebraic Multigrid (AMG) for Saddle Point Systems*. PhD thesis, Institut für Numerische Simulation, Universität Bonn, July 2013.
- [27] S. Milewski. Meshless finite difference method with higher order approximation—applications in mechanics. *Archives of Computational Methods in Engineering*, 19(1):1–49, 2012.
- [28] S. Milewski. Selected computational aspects of the meshless finite difference method. *Numerical Algorithms*, 63(1):107–126, 2013.

- [29] A. Möller and J. Kuhnert. Simulation of the glass flow inside a floating process / Simulation de l'écoulement du verre dans le procédé float. *Revue Verre*, 13(5):28–30, 2007.
- [30] E. Oñate, S. Idelsohn, O. C. Zienkiewicz, and R. L. Taylor. A finite point method in computational mechanics. applications to convective transport and fluid flow. *International Journal for Numerical Methods in Engineering*, 39(22):3839–3866, 1996.
- [31] S. Patankar and D. Spalding. A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *International Journal of Heat and Mass Transfer*, 15(10):1787 – 1806, 1972.
- [32] C. Praveen and S. M. Deshpande. Kinetic meshless method for compressible flows. *International Journal for Numerical Methods in Fluids*, 55(11):1059–1089, 2007.
- [33] E. O. Reséndiz-Flores and F. R. Saucedo-Zendejo. Two-dimensional numerical simulation of heat transfer with moving heat source in welding using the finite pointset method. *International Journal of Heat and Mass Transfer*, 90:239 – 245, 2015.
- [34] B. Seibold. *M-Matrices in Meshless Finite Difference Methods*. PhD thesis, Kaiserslautern University, 2006.
- [35] F. S. Sousa, C. M. Oishi, and G. C. Buscaglia. Spurious transients of projection methods in microflow simulations. *Computer Methods in Applied Mechanics and Engineering*, 285:659 – 693, 2015.
- [36] D. Sridar and N. Balakrishnan. An upwind finite difference scheme for meshless solvers. *Journal of Computational Physics*, 189(1):1 – 29, 2003.
- [37] P. Suchde, J. Kuhnert, S. Schröder, and A. Klar. A flux conserving meshfree method for conservation laws. In Press, 2016.
- [38] S. Tiwari and J. Kuhnert. Finite pointset method based on the projection method for simulations of the incompressible navier-stokes equations. In M. Griebel and M. A. Schweitzer, editors, *Meshfree Methods for Partial Differential Equations*, pages 373–387. Springer, 2002.
- [39] S. Tiwari and J. Kuhnert. A meshfree method for incompressible fluid flows with incorporated surface tension. *Revue Européenne des Éléments*, 11(7-8):965–987, 2002.
- [40] A. Tramecon and J. Kuhnert. Simulation of advanced folded airbags with VPS-PAMCRASH/FPM: Development and validation of turbulent flow numerical simulation techniques applied to curtain bag deployments. In *SAE Technical Paper*. SAE International, 2013.
- [41] H. A. van der Vorst. Bi-cgstab: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 13(2):631–644, 1992.
- [42] R. Xu, P. Stansby, and D. Laurence. Accuracy and stability in incompressible SPH (ISPH) based on the projection method and a new approach. *Journal of Computational Physics*, 228(18):6703 – 6725, 2009.
- [43] Q. Zhang. A fourth-order approximate projection method for the incompressible navierstokes equations on locally-refined periodic domains. *Applied Numerical Mathematics*, 77:16 – 30, 2014.
- [44] T. Zhang, Y.-F. Ren, C.-M. Fan, and P.-W. Li. Simulation of two-dimensional sloshing phenomenon by generalized finite difference method. *Engineering Analysis with Boundary Elements*, 63:82 – 91, 2016.
- [45] D. Zhou, B. Seibold, D. Shirokoff, P. Chidyagwai, and R. R. Rosales. Meshfree finite differences for vector poisson and pressure poisson equations with electric boundary conditions. In M. Griebel and M. A. Schweitzer, editors, *Meshfree Methods for Partial Differential Equations VII*, pages 223–246, Cham, 2015. Springer International Publishing.